(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 0 903 678 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
24.03.1999 Bulletin 1999/12

(51) Int. Cl.[6]: **G06F 17/60**

(21) Application number: **98117821.3**

(22) Date of filing: **19.09.1998**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **23.09.1997 GB 9720166**

(71) Applicant:
International Computers Ltd.
London, EC2A 1DS (GB)

(72) Inventors:
- **Parker, Jeffrey Martin**
  Barlaston, Staffordshire (GB)
- **Westmacott, Peter**
  Sandbach, Cheshire CW11 1EJ (GB)

(74) Representative:
Guyatt, Derek Charles et al
Intellectual Property Department
International Computers Limited
Cavendish Road
Stevenage, Herts, SG1 2DY (GB)

(54) Workflow management system

(57) A workflow management system consists of a number of work pool objects, holding work items waiting to be processed, and a number of work pool user objects, for processing work items from the work pool objects in accordance with rules stored in the individual work pool user objects. Each of the work pool objects has the capability of managing its own work pool user objects, and has a number of work pool viewer objects associated with it, for allowing respective users to view and select the work items held by that work pool object. When a work item is selected by a user, the work pool viewer object informs the associated work pool object, which then sends a copy of the selected work item to the appropriate work pool user object. A system administrator object allowing a user to modify the business rules in any selected work pool user object dynamically, while the system is running.

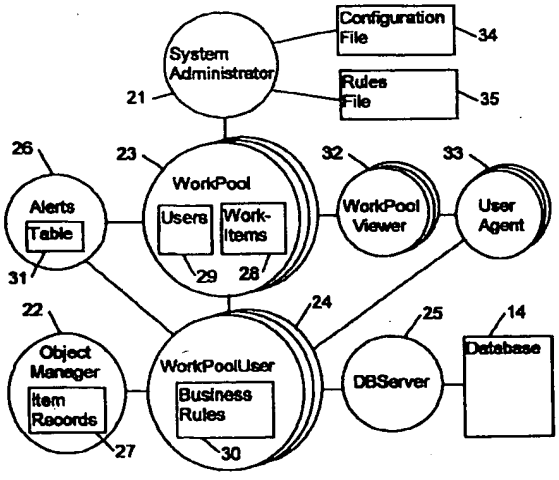

FIG.2

EP 0 903 678 A2

Description

Background to the Invention

[0001] This invention relates to workflow management systems.

[0002] A workflow management system is a computer-implemented system for providing support for business processes. For example, such a system may be used in a manufacturing business, to support the process of handling orders from customers. Typically, such a system maintains a work pool, containing work items (such as customer orders) waiting to be processed, and allows a number of individual users (such as sales, accounts and purchasing personnel) to access the work items in the pool.

[0003] Conventional workflow management systems of this kind contain a central set of rules, which define the particular business process being implemented. For example, these rules may define which users should handle each work item, the operations that may be performed by each particular user, and the further work items to be generated.

[0004] A problem with this is that it is difficult to make changes to the system, such as changing the sequence of processing of work items or introducing new users into the system. It is particularly difficult to make such changes while the system is actually in use.

[0005] The object of the present invention is to provide a novel form of workflow management system in which the problem of making changes to the system is simplified.

Summary of the Invention

[0006] According to the invention a workflow management system comprises:

(a) a plurality of work pool objects, for holding work items waiting to be processed, and
(b) a plurality of work pool user objects, for processing work items from the work pool objects in accordance with rules stored in the individual work pool user objects.

[0007] The fact that the rules are stored in the work pool user objects themselves, rather than centrally, makes it much easier to modify or reconfigure the system, even while the system is running.

Brief Description of the Drawings

[0008]

Figure 1 is a block diagram of a computer system incorporating a workflow management system.

Figure 2 is a block diagram showing the workflow management system in more detail.

Description of an Embodiment of the Invention

[0009] One workflow management system in accordance with the invention will now be described by way of example with reference to the accompanying drawings.

Overall view of the system

[0010] Figure 1 shows a computer system comprising a server computer 10, and a number of user terminals 11 connected to the server by way of a network 12. The server runs a workflow management system 13, which can access a database 14.

[0011] The work of the system is divided into units referred to herein as cases, which are processed independently of each other. For example, a case may correspond to a customer order. When a new case is introduced into the system, a data item, referred to herein as a business object, is created in the database. The business object contains all the business data relevant to that case (for example, order number, customer name, quantity, price and so on). The content of business objects may vary, and different business objects may be held in different database tables. A case may have a number of business objects associated with it, logically organised in a tree structure in which each object may have one or more subsidiary (child) objects. For example, a business object representing a customer order may have subsidiary business objects representing associated internal purchase orders.

[0012] Typically, each case will require a number of operations to be performed on the associated business object, by users or external tools. This is achieved by placing work items in a number of work pools. Each business object may have a number of work items associated with it, allowing a number of operations to be performed on a case in parallel. Users can view the contents of the work pools, and select work items for processing, according to predefined business rules. This processing may, in turn, generate further work items for the work pools.

Workflow management system

[0013] Figure 2 shows the workflow management system 13 in more detail.

[0014] The workflow management system comprises a number of active processing objects referred to herein as roles, each of which encapsulates its local data and represents an independent thread of execution. These roles include a SystemAdministrator role 21, an ObjectManager role 22, a number of WorkPool roles 23, a number of WorkPoolUser roles 24, a DBServer role 25, and an Alerts role 26. These various roles may conveniently be built using the role based process technology provided by the ICL Processwise Integrator, as

described for example in the following reference:

"The Use of a Persistent Language in the Implementation of a Process Support System"
R.Mark Greenwood et al.
ICL Technical Journal, May 1992, page 109.

[0015] The SystemAdministrator role 21 is responsible for the introduction of users to the system, and for the creation, deletion and modification of the other roles.

[0016] The ObjectManager role 22 is responsible for managing the business objects. This role holds an internal table 27 containing a number of item records, one for each business object held in the database. Whenever a business object is created, a corresponding item record is also created, for referencing the business object. The item record contains information about the business object in a standard format. The tree structure of the business objects is represented by parent-child links in the corresponding item records. The ObjectManager role also provides a locking mechanism for locking the business objects, so as to allow only one user at a time to update a given object.

[0017] The WorkPool roles 23 are responsible for maintaining the work pools. One instance of the WorkPool role is created for each work pool in the system. Each WorkPool role instance holds an internal table 28 containing a set of work items, which form the work pool. It also holds an internal table 29, containing details of the users who are assigned to this work pool. For each user, the user table holds the user's full name, login name and password.

[0018] The WorkPoolUser roles 24 are responsible for processing the work items. One instance of the WorkPoolUser role is created for each user assigned to each work pool. In other words, each WorkPool role has a number of WorkPoolUser roles associated with it, one for each user assigned to that work pool, and each user has one WorkPoolUser role for each work pool that can be accessed by that user. A WorkPoolUser role may be associated with an external application rather than (or as well as) a human operator. Each WorkPoolUser role instance holds a set of business rules tables 30, specifying how this role is to handle work items, including when information about when to pass work to other roles.

[0019] The DBServer role 25 provides a common point of access to the database 14 for the other roles.

[0020] The Alerts role 26 handles deadlines for processing work items. It contains an internal table 31. In response to a registerAlert request from a WorkPoolUser role, it creates a record in this internal table, specifying the deadline date and time, and the work item to which the deadlin relates. The Alerts role periodically monitors this table, to check whether any deadlines have expired. When a deadline expires, the Alerts role sends a timerEvent call to the appropriate WorkPool role, which in turn calls the appropriate WorkPoolUser role to handle the alert.

[0021] The system also includes a number of WorkPoolViewer objects 32, one for each user assigned to each work pool. Each WorkPoolViewer object enables a user to access a work pool, to view a list of all the work items currently in that work pool, and to select one of the work items for viewing or for action.

[0022] The system also includes a number of user agents 33, which provide interfaces between the workflow management system and the respective users.

Item records

[0023] As mentioned above, the ObjectManager role holds an item record for each business object in the database. The structure of an item record is as follows:

id	The record identifier. This is the same as the database key of the corresponding business object.
type	The name of the database table in which the business object is held.
partOf	The identifier of the parent item record (if any).
children	A list of the identifiers of all the child item records (if any) for this record.
caseId	The identifier of the topmost parent of this family of item records.
summary	A description field.
workItems	A table of the work items associated with this business object. For each work item, the table contains a copy of the work item, the identity of the work pool in which the item is located, and the work pool type.

Work items

[0024] As mentioned above, each WorkPool role holds a number of work items, constituting a work pool. The structure of each work item is as follows:

id	Identifier. This is a reference, local to the work pool, which identifies the work item.
objID	Object identifier. This is a reference to the corresponding item record.
summary	A copy of the summary field in the corresponding item record.
workType	A copy of the type field in the corresponding item record.
priority	Indicates whether or not the user is required to take action on this work item.
date	Deadline date.
assignedTo	The name of the user to whom the work item is currently assigned.
state	The current state of the work item.

[0025] As will be described, the "state" field of a work

item determines the operations that are to be performed on it when it is selected by a user.

WorkPoolViewer

[0026] Whenever a user logs in to the system, that user's WorkPoolViewer sends a message to the associated WorkPool role, requesting it to supply a copy of its work pool. In response to this request, the WorkPool role copies its current set of work items to the WorkPoolViewer, where the items are displayed to the user. The displayed information for each work item is derived from the objID, summary, workType, state, priority, date, and assignedTo fields. As will be described, whenever th WorkPool role changes the content of any work item, it notifies all its associated WorkPoolViewers of the change, so that they can keep their displayed information up to date.

[0027] The WorkPoolViewer allows the user to select any one of the displayed work items, for example by clicking on the item with a mouse. The user may then either accept the selected work item, or simply request to view it. If the user accepts an item, the WorkPoolViewer sends an "accept" message to the WorkPool role. Alternatively, if the user requests to view an item, the WorkPoolViewer sends an "view" message to the WorkPool role. Both of these messages contain the identity of the selected work item.

WorkPool role

[0028] When a WorkPool role receives an "accept" message from a WorkPoolViewer, it performs the following operations. The WorkPool role first checks whether the selected work item is available for assignment, i.e. whether the assignedTo field of the work item is empty. If so, the assignedTo field is set to the login name of the user, and the WorkPool role then sends a manageWorkItem request to the WorkPoolUser role for this user. The request contains a copy of the work item, and an "assign" command. The WorkPool role also sends a copy of the updated work item to all its associated WorkPoolViewers, to notify them of this change.

[0029] If the WorkPool role receives a "view" message from a WorkPoolViewer, it performs the following operations. The WorkPool role sends a manageWorkItem request to WorkPoolUser role, containing a copy of the work item, and a "view" command. In this case the WorkPool role does not check or set the assignedTo field.

[0030] The WorkPool role can also receive a routeWorkItem request from any of the WorkPoolUser roles, specifying one of the following commands: "add", "replace", "release" and "delete". In response to an "add" command, the WorkPool role creates a new work item with a unique identifier in its internal table, and returns the identifier of the new work item to the calling WorkPoolUser role. In response to a "replace" command, the WorkPool role replaces an identified work item in its internal tables with an updated version of the work item, and sets the assignedTo field of the work item to the empty string. In response to a "release" command, the WorkPool role sets the assignedTo field of an identified work item to the empty string. In response to a "delete" command, the WorkPool role deletes an identified work item from its internal table. In each case, the WorkPool role notifies all the associated WorkPoolViewers of the change. In the case of the "replace", "release" and "delete" commands, if the identified work item is not present, the WorkPool role returns an error message to the caller.

[0031] The WorkPool role can also receive a timerEvent call from the Alerts role, to notify it that a deadline has expired in relation to a specified work item. In response to this call, the Workpool role locks the specified work item (on behalf of an arbitrarily chosen user), and then sends a manageWorkItem request to the WorkPoolUser role. The request contains the contents of the work item, and an "alert" command. If the identified work item is not present in the work pool, the alert is ignored.

WorkPoolUser role

[0032] When a WorkPoolUser role receives a manageWorkItem request from a WorkPool role, it performs the following actions.

[0033] The WorkPoolUser role first calls the ObjectManager role, requesting it fetch and lock the item record identified by the objID field of the work item. If the item record is not locked, the ObjectManager role will lock it and return a copy of the item record, including all the related work items. However, if the item record is already locked, an error message is returned.

[0034] The WorkPoolUser role then accesses its locally held business rules tables, using the "state" field of the work item as a key, to obtain the following information:

- A form to be displayed to the user.
- An explanatory message for the user.
- An action list for the user to select from.

[0035] The WorkPoolUser role also calls the DBServer role to fetch business data from the business object referenced by the item record, and also to fetch process data relevant to the current state.

[0036] The WorkPoolUser role then calls its associated user agent, requesting it to display the form, message and action list, with the data retrieved from the database in appropriate fields of the form. The user may then enter data, update fields, and select actions in the form. The form includes "OK" and "Cancel" buttons. If the user presses the "OK" button, the user agent returns the updated fields to the WorkPoolUser role, along with an "action" field specifying th action selected by the

user.

[0037] When the WorkPoolUser role receives this information, it concatenates the "step" field of the work item with the "action" field returned by the user, to form a key. This key is used to access a nextSteps table in the business rules, so as to obtain a series of process steps to be carried out.

[0038] Each process step may involve creating a new work item in a target work pool, or replacing or deleting an existing work item in a target work pool. Creating, replacing or deleting a work item is performed by sending a routeWorkItem request containing an "add", "replace" or "delete" command to the appropriate WorkPool role. The target work pool may be the current work pool, or may be another specified work pool. When a work item is created, replaced or deleted, a request also is sent to the ObjectManager role, requesting it to update the list of work items in the corresponding item record. If the process step specifies a deadline for the work item, a registerAlert request is sent to the Alerts role, so as to set up a new deadline.

[0039] The particular sequence of steps depends on the particular business process being supported by the system. For example, when a work item representing a new sale is processed by a salesperson, the business rules may specify that a further work item, representing an invoicing task, should be created and placed in another work pool for access by an accounts clerk.

[0040] Finally, any business objects affected are updated by passing the object identifier, type and data table to the DBServer role, and the corresponding item records are updated by calling the ObjectManager role.

[0041] The WorkPoolUser role is then ready to accept another manageWorkItem request.

[0042] If permitted by its business rules, a WorkPoolUser role may also allow its user to create a new case, e.g. a salesperson may be permitted to create a new customer order, while an accounts clerk would not.

SystemAdministrator

[0043] When the workflow management system is installed, the SystemAdministrator role automatically customises the system for the particular business process being supported. The information for customising the system is supplied in a set of configuration files 34 and business rules files 35. The configuration files specify the required work pools and the users that are to be assigned to each of these work pools. The business rules files specify the business rules for the system.

[0044] The SystemAdministrator role first reads the configuration files, and creates a user agent object 33 for each user. It also creates a WorkPool role 23 for each required work pool, with the names of its users held in its internal table. The WorkPool roles are created as instances of a general WorkPool type.

[0045] Each WorkPool role instance then calls the SystemAdministrator role 21, requesting it to create a WorkPoolUser role 24 for each of its users. Each WorkPoolUser role is given a copy of the business rules relevant to its part in the behaviour of the overall process. The WorkPoolUser roles are created as instances of a general WorkPoolUser type. A WorkPoolViewer object 32 is also created for each user.

[0046] The SystemAdministrator role can also be called at any time, by a suitably authorised user, to modify the business rules in any WorkPoolUser role, to create new work pools, to introduce new users to the work pools, or to remove existing users from the work pools. For example, specialised roles can readily be created to cater for special requirements. These actions can be performed dynamically, while the application is running. This ability to dynamically modify the system is a consequence of the fact that each of the WorkPool roles stores its own internal list of users, and each of the WorkPoolUser roles stores its own internal business rules.

Some possible modifications

[0047] It will be appreciated that many modifications may be made to the system described above without departing from the scope of the present invention.

Claims

1. A workflow management system comprising:

   (a) a plurality of work pool objects, for holding work items waiting to be processed, and
   (b) a plurality of work pool user objects, for processing work items from the work pool objects in accordance with rules stored in the individual work pool user objects.

2. A workflow management system according to Claim 1 wherein each of the work pool objects has the capability of managing its own work pool user objects.

3. A workflow management system according to Claim 1 or 2 wherein each of the work pool objects has a number of work pool viewer objects associated with it, for allowing respective users to view the work items held by that work pool object and to select one of those work items.

4. A workflow management system according to Claim 3 wherein, when a work item is selected by a user through a work pool viewer object, the work pool viewer object informs the associated work pool object, and the work pool object then sends a copy of the selected work item to the work pool user object associated with that user.

5. A workflow management system according to any

preceding claim, including a system administrator object for allowing a user to modify the business rules in any selected work pool user object dynamically, while the system is running.

6. A method of workflow management in a computer system comprising the steps:

(a) maintaining a plurality of work pool objects, for holding work items waiting to be processed, and
(b) maintaining a plurality of work pool user objects, for processing work items from the work pool objects in accordance with rules stored in the individual work pool user objects.

7. A method according to Claim 6 wherein each of the work pool objects has the capability of managing its own work pool user objects.

8. A method according to Claim 6 or 7 wherein each of the work pool objects has a number of work pool viewer objects associated with it, for allowing respective users to view the work items held by that work pool object and to select one of those work items.

9. A method according to Claim 8 wherein, when a work item is selected by a user through a work pool viewer object, the work pool viewer object informs the associated work pool object, and the work pool object then sends a copy of the selected work item to the work pool user object associated with that user.

10. A method according to any one of Claims 6 to 9, including the further step of using a system administrator object to modify the business rules in any selected work pool user object dynamically, while the system is running.

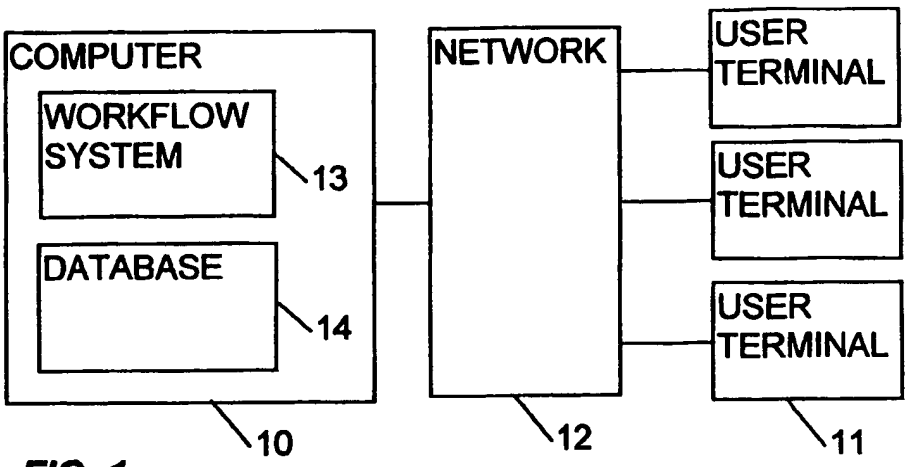
COMPUTER
WORKFLOW SYSTEM
13
DATABASE
14
10
NETWORK
12
USER TERMINAL
USER TERMINAL
USER TERMINAL
11

FIG. 1

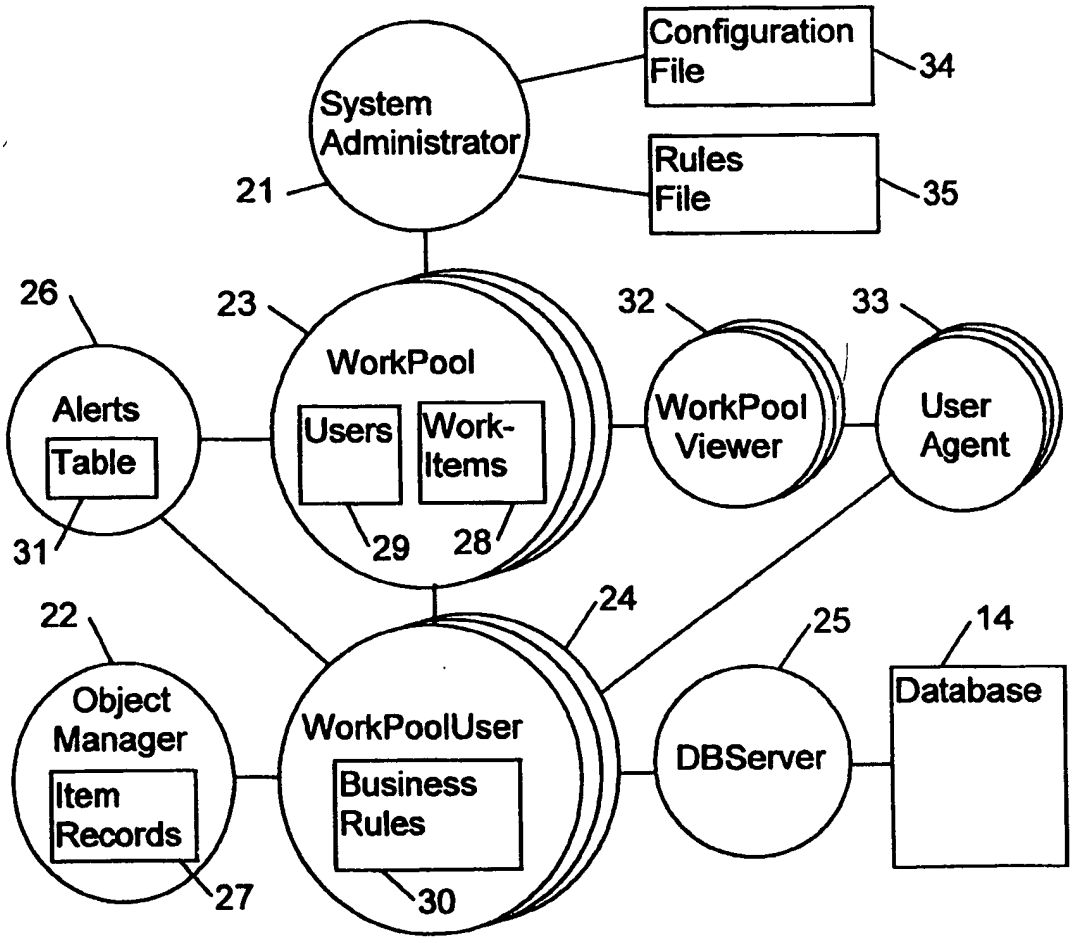
System Administrator
21
Configuration File
34
Rules File
35
26
23
WorkPool
Users
Work-Items
29
28
32
WorkPool Viewer
33
User Agent
Alerts
Table
31
22
Object Manager
Item Records
27
24
WorkPoolUser
Business Rules
30
25
DBServer
14
Database

FIG.2